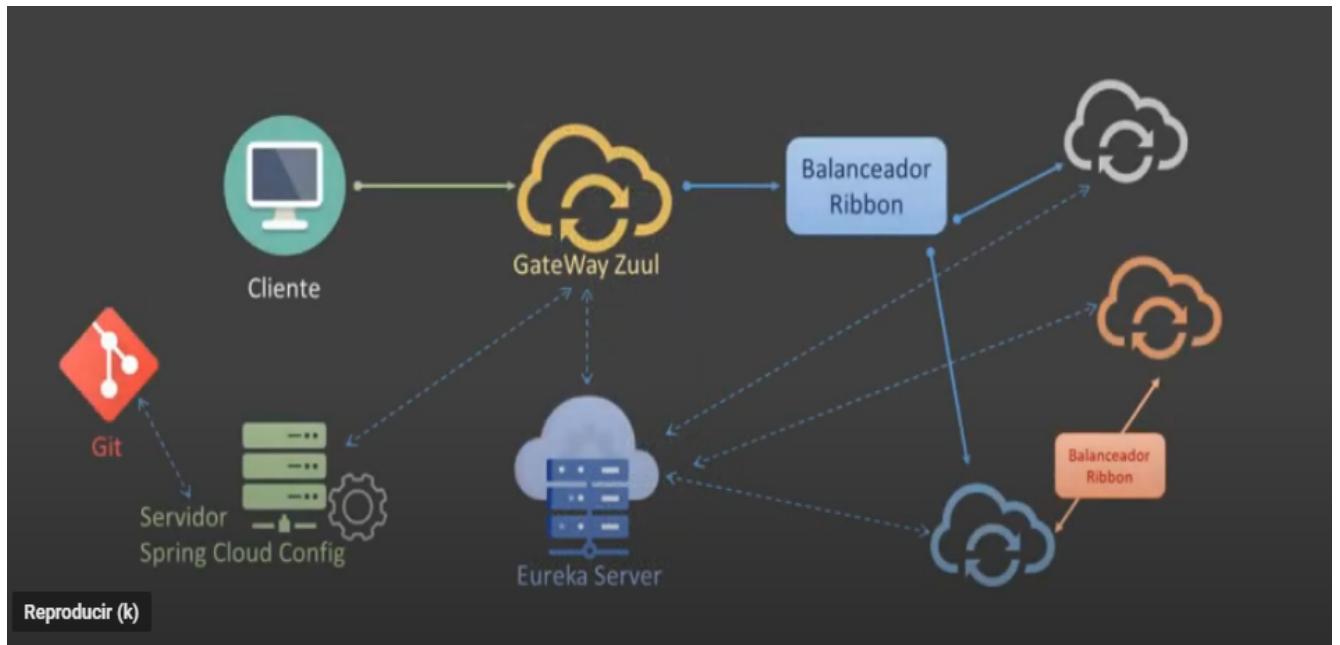


# Ecosistema

La estructura inicial del ecosistema es la siguiente



Cliente :Front end en Angular PrimeFaces

Zuul: Es la puerta de entrada al ecosistema ,se puede usar spring cloud gateway

Ribbon: Es el balanceador de carga.

Spring Cloud Config: servidor de configuración de las propiedades de los microservicios

Eureka: Servidor de microservicios.

## Configuración Eureka

SpringbootServicioEurekaServerApplication.java

```
SpringbootServicioEurekaServerApplication.java X
1 package com.formacionbdi.springboot.app.eureka;
2
3 import org.springframework.boot.SpringApplication;
4
5 @EnableEurekaServer
6 @SpringBootApplication
7 public class SpringbootServicioEurekaServerApplication {
8
9     public static void main(String[] args) {
10         SpringApplication.run(SpringbootServicioEurekaServerApplication.class, args);
11     }
12 }
```

## pom.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
4   <modelVersion>4.0.0</modelVersion>
5   <parent>
6     <groupId>org.springframework.boot</groupId>
7     <artifactId>spring-boot-starter-parent</artifactId>
8     <version>2.5.3</version>
9     <relativePath/> <!-- lookup parent from repository -->
10    </parent>
11    <groupId>com.formacionbdi.springboot.app.eureka</groupId>
12    <artifactId>springboot-servicio-eureka-server</artifactId>
13    <version>0.0.1-SNAPSHOT</version>
14    <name>springboot-servicio-eureka-server</name>
15    <description>Demo project for Spring Boot</description>
16
17    <properties>
18      <java.version>12</java.version>
19      <spring-cloud.version>2020.0.3</spring-cloud.version>
20    </properties>
21
22    <dependencies>
23      <dependency>
24        <groupId>org.glassfish.jaxb</groupId>
25        <artifactId>jaxb-runtime</artifactId>
26      </dependency>
27
28      <dependency>
29        <groupId>org.springframework.cloud</groupId>
30        <artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>
31      </dependency>
32
33    </dependencies>

```

## applications.properties

- Spring Tool Suite 4

```
1 spring.application.name=servicio-eureka-server
2 server.port=8761
3
4 eureka.client.register-with-eureka=false
5 eureka.client.fetch-registry=false
```

Para que los microservicios se registren en eureka como clientes con puertos dinámicos se deben configurar de la siguiente manera.

El ejemplo es del microServicio producto.

'productos/SpringbootServicioProductosApplication.java - Spring Tool Suite 4

The screenshot shows the Spring Tool Suite 4 interface with three tabs open: `SpringbootServicioProductosApplication.java`, `springboot-servicio-productos/pom.xml`, and `application.properties`.

`SpringbootServicioProductosApplication.java` content:

```
1 package com.formacionbdi.springboot.app.productos;
2
3 import org.springframework.boot.SpringApplication;
4
5 @EnableEurekaClient
6 @SpringBootApplication
7 public class SpringbootServicioProductosApplication {
8
9     public static void main(String[] args) {
10         SpringApplication.run(SpringbootServicioProductosApplication.class, args);
11     }
12 }
```

`pom.xml` content (with a red box highlighting the Eureka dependency section):

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
</dependency>

<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-devtools</artifactId>
    <scope>runtime</scope>
</dependency>
<dependency>
    <groupId>com.h2database</groupId>
    <artifactId>h2</artifactId>
    <scope>runtime</scope>
</dependency>
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
</dependency>
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
</dependency>
</dependencies>
```

`application.properties` content:

```
1 spring.application.name=servicio_productos
2 #Para tener diferentes instancias en puertos aleatorios en Eureka
3 server.port=${PORT:0}
4 eureka.instance.instance-id=${spring.application.name}:${spring.application.instance_id:${random.value}}
5 #Ubicación de Eureka
6 eureka.client.service-url.defaultZone=http://localhost:8761/eureka
```

Tool Suite 4

The screenshot shows the Spring Tool Suite 4 interface with three tabs open: `SpringbootServicioProductosApplication.java`, `springboot-servicio-productos/pom.xml`, and `application.properties`.

`application.properties` content (with a red box highlighting the Eureka configuration section):

```
1 spring.application.name=servicio_productos
2 #Para tener diferentes instancias en puertos aleatorios en Eureka
3 server.port=${PORT:0}
4 eureka.instance.instance-id=${spring.application.name}:${spring.application.instance_id:${random.value}}
5 #Ubicación de Eureka
6 eureka.client.service-url.defaultZone=http://localhost:8761/eureka
```

## Spring Cloud Config

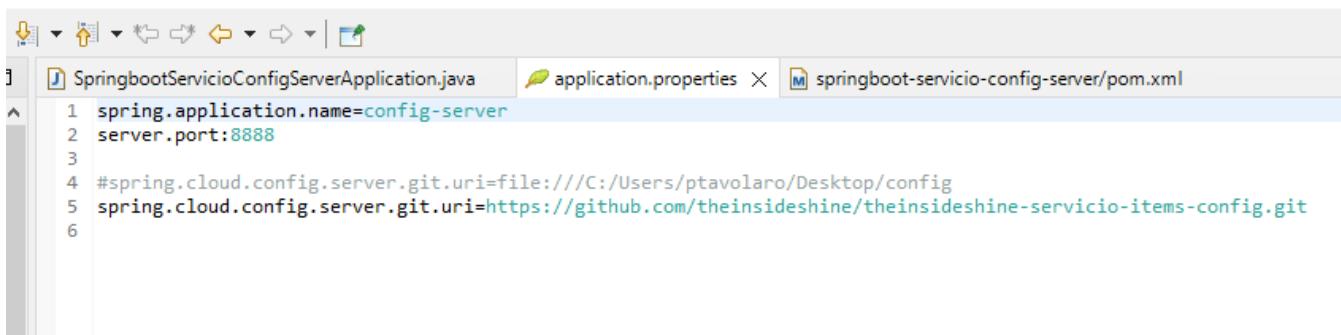
Las configuraciones como variables de entorno, pueden almacenarse en un servidor de configuración que está conectado a un repositorio, uno de los principales objetivos es poder cambiar la configuración sin reiniciar los microservicios.

.ios/app/config/SpringbootServicioConfigServerApplication.java - Spring Tool Suite 4

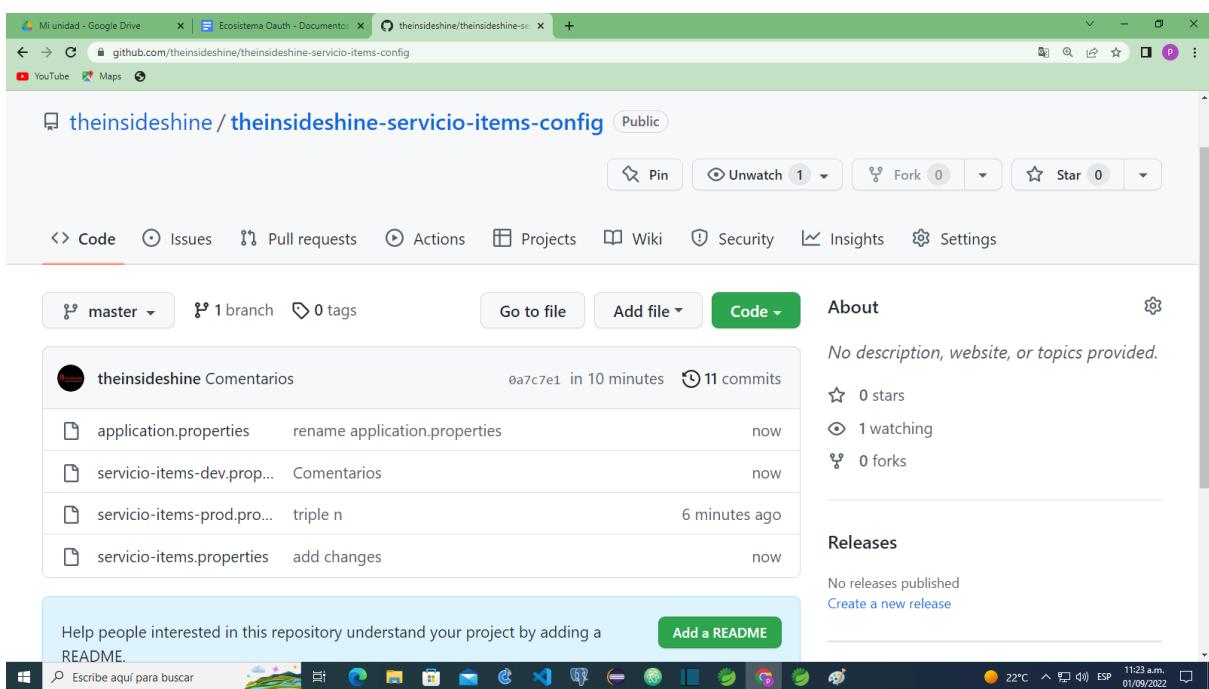
```
SpringbootServicioConfigServerApplication.java X application.properties M springboot-servicio-config-server/pom.xml
1 package com.formacionbdi.microservicios.app.config;
2
3④ import org.springframework.boot.SpringApplication;
4
5 @EnableConfigServer
6 @SpringBootApplication
7 public class SpringbootServicioConfigServerApplication {
8
9     public static void main(String[] args) {
10         SpringApplication.run(SpringbootServicioConfigServerApplication.class, args);
11     }
12 }
13
14 }
```

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>2.7.3</version>
    <relativePath/> <!-- lookup parent from repository -->
  </parent>
  <groupId>com.formacionbdi.microservicios.app.config</groupId>
  <artifactId>springboot-servicio-config-server</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <name>springboot-servicio-config-server</name>
  <description>Demo project for Spring Boot</description>
  <properties>
    <java.version>12</java.version>
    <spring-cloud.version>2021.0.3</spring-cloud.version>
  </properties>
  <dependencies>
    <dependency>
      <groupId>org.springframework.cloud</groupId>
      <artifactId>spring-cloud-config-server</artifactId>
    </dependency>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-devtools</artifactId>
      <scope>runtime</scope>
      <optional>true</optional>
    </dependency>
```

#### Spring Tool Suite 4



```
1 spring.application.name=config-server
2 server.port:8888
3
4 #spring.cloud.config.server.git.uri=file:///C:/Users/ptavolaro/Desktop/config
5 spring.cloud.config.server.git.uri=https://github.com/theinsidesshine/theinsidesshine-servicio-items-config.git
6
```



theinsidesshine / theinsidesshine-servicio-items-config (Public)

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

master 1 branch 0 tags Go to file Add file Code

**theinsidesshine** Comentarios 0a7c7e1 in 10 minutes 11 commits

application.properties rename application.properties now

servicio-items-dev.prop... Comentarios now

servicio-items-prod.pro... triple n 6 minutes ago

servicio-items.properties add changes now

About

No description, website, or topics provided.

0 stars 1 watching 0 forks

Releases

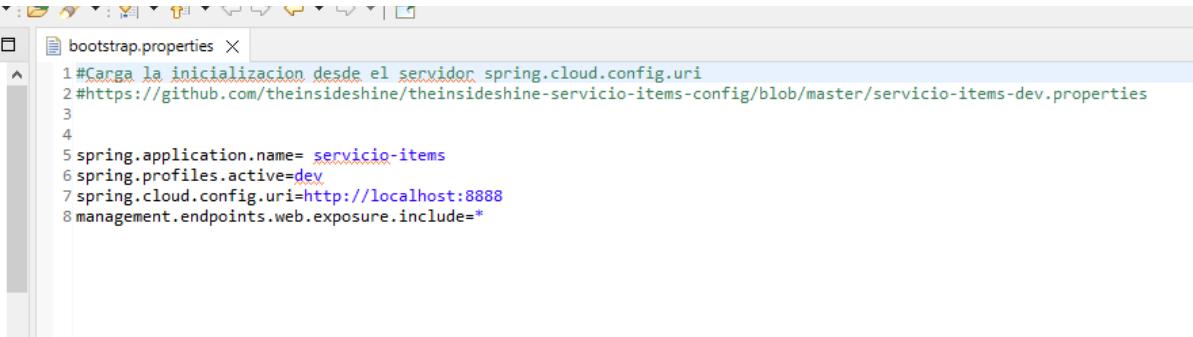
No releases published Create a new release

para actualizar la configuración acceder a la siguiente ruta:

POST localhost:port{Microservicio}/actuator/refresh

Para activar la configuración en cada microservicio se puede utilizar el archivo bootstrap.properties , donde localhost:8888 es el endpoint de servidor de configuración

El ejemplo es del microServicio items.



A screenshot of a code editor window titled "bootstrap.properties". The content of the file is as follows:

```
1 #Carga la inicializacion desde el servidor spring.cloud.config.uri
2 https://github.com/theinsideshine/theinsideshine-servicio-items-config/blob/master/servicio-items-dev.properties
3
4
5 spring.application.name= servicio-items
6 spring.profiles.active=dev
7 spring.cloud.config.uri=http://localhost:8888
8 management.endpoints.web.exposure.include=*
```

GateWay: zuul ,desarrollado por netflix, está en modo mantenimiento.

Deben configurarse los nombres de todos los microservicios ,los timeout de los intentos por fallas.

```
SpringbootServicioZuulServerApplication.java X application.properties M springboot-servicio-zuul-server/pom.xml
1 package com.formacionbdi.springboot.app.zuul;
2
3 import org.springframework.boot.SpringApplication;
4
5 @EnableEurekaClient
6 @EnableZuulProxy
7 @SpringBootApplication
8 public class SpringbootServicioZuulServerApplication {
9
10     public static void main(String[] args) {
11         SpringApplication.run(SpringbootServicioZuulServerApplication.class, args);
12     }
13 }
```

```
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
</dependency>
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-netflix-zuul</artifactId>
</dependency>
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-devtools</artifactId>
    <scope>runtime</scope>
    <optional>true</optional>
</dependency>
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
</dependency>
<dependency>
    <groupId>org.springframework.security.oauth</groupId>
    <artifactId>spring-security-oauth2</artifactId>
    <version>2.3.8.RELEASE</version>
</dependency>
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-jwt</artifactId>
    <version>1.1.1.RELEASE</version>
</dependency>
<dependency>
    <groupId>org.glassfish.jaxb</groupId>
    <artifactId>jaxb-runtime</artifactId>
</dependency>
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-config</artifactId>
</dependency>
```

#### pring Tool Suite 4

The screenshot shows the Spring Tool Suite 4 interface with three tabs open: `SpringbootServicioZuulServerApplication.java`, `application.properties`, and `springboot-servicio-zuul-server/pom.xml`. The `application.properties` tab contains the following configuration:

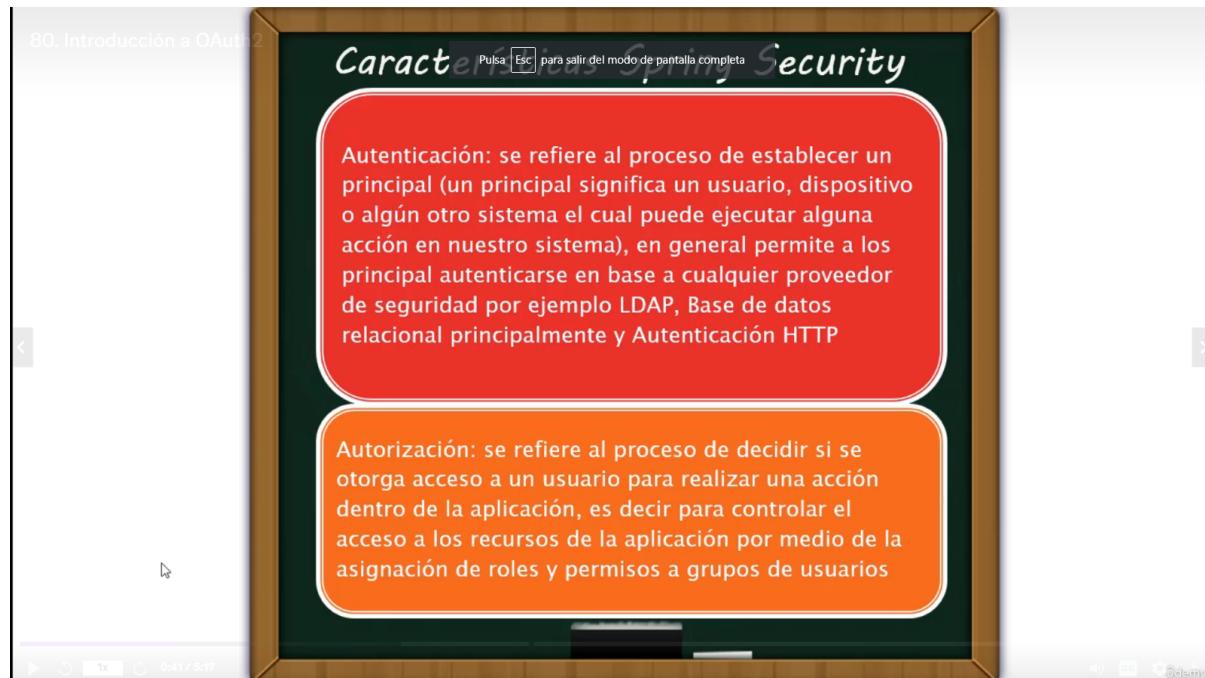
```
1 spring.application.name=servicio-zuul-server
2 server.port=8090
3
4 #servidor Eureka
5 eureka.client.serviceUrl.defaultZone=http://localhost:8761/eureka/
6
7 #microServicios
8
9 zuul.routes.productos.service-id=servicio-productos
10 zuul.routes.productos.path=/api/productos/**
11
12 zuul.routes.items.service-id=servicio-items
13 zuul.routes.items.path=/api/items/**
14
15 zuul.routes.usuarios.service-id=servicio-usuarios
16 zuul.routes.usuarios.path=/api/usuarios/**
17
18 zuul.routes.security.service-id=servicio-oauth
19 zuul.routes.security.path=/api/security/**
20 zuul.routes.security.sensitive-headers=Cookie,Set-Cookie
21
22
23 #timeOuts para sistema de fallos
24 hystrix.command.default.execution.isolation.thread.timeoutInMilliseconds: 20000
25 ribbon.ConnectTimeout: 3000
26 ribbon.ReadTimeout: 10000
```

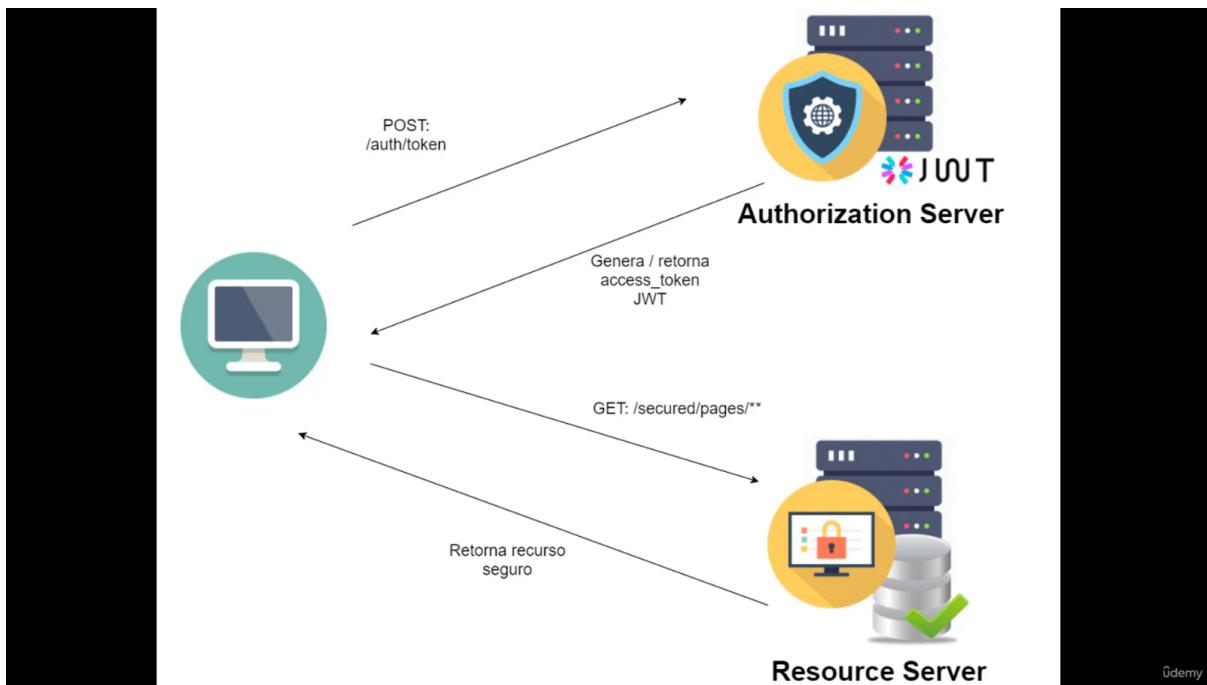
## Spring security

las dos partes principales de estas separadas

MicroServicio Oauth: autenticacion

Zull: Autorizacion.





Udemy

## url: POST /auth/token

### header:

- **Authorization: Basic Base64(client\_id:client\_secret)**
- **Content-Type: application/x-www-form-urlencoded**

### body:

- **grant\_type = password**
- **username = andres**
- **password = 12345**



{

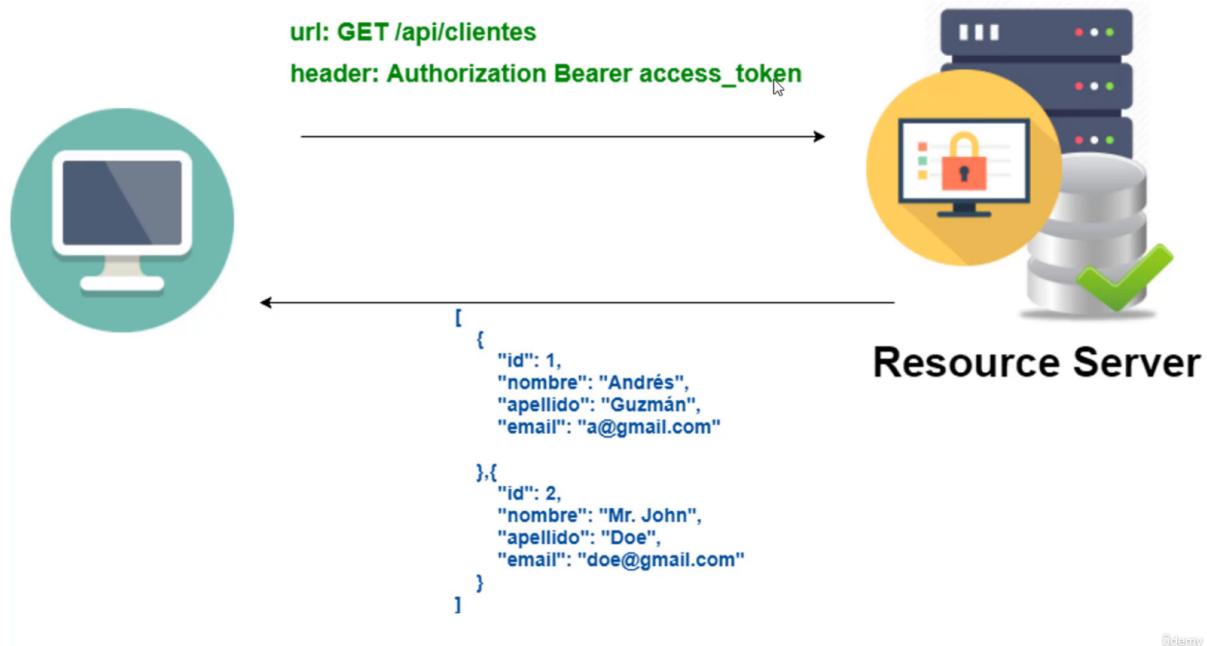
```

    "access_token": "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9...",
    "token_type": "bearer",
    "refresh_token": "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9...",
    "expires_in": 3599,
    "scope": "read write",
    "jti": "58efb674-46e6-4f6b-bbf0-e92e21e4b34a"
  }
```



**Authorization Server**

Udemy



Cómo comparten las claves de jwt los el gateway Zuul y el Ms oauth, se configura el uso de el servidor de configuración en los dos.  
para esto se usa el archivo bootstrap.properties

IntelliJ 4

```

bootstrap.properties
1 spring.application.name= servicio-oauth
2 spring.cloud.config.uri=http://localhost:8888
3 management.endpoints.web.exposure.include=*
  
```

Idea Suite 4

```

bootstrap.properties
1 spring.application.name= servicio-zuul-server
2 spring.cloud.config.uri=http://localhost:8888
3 management.endpoints.web.exposure.include=*
  
```

master theinsideshine-servicio-items-config / application.properties

theinsideshine rename application.properties

Latest commit

1 contributor

3 lines (3 sloc) | 142 Bytes

```
1 config.security.oauth.client.id=frontendapp
2 config.security.oauth.client.secret=12345
3 config.security.oauth.jwt.key=algun_codigo_secreto_aeiou
```

## MicroServicio Oauth: autenticacion

Debe incluirse las entidades alumnos, las dependencias para la comunicación entre microservicios y las de SpringSecurity

pom.xml

<https://github.com/theinsideshine/microservicios-oauth/blob/main/springboot-servicio-oauth/pom.xml>

application.properties

<https://github.com/theinsideshine/microservicios-oauth/blob/main/springboot-servicio-oauth/src/main/resources/application.properties>

appStart.java

<https://github.com/theinsideshine/microservicios-oauth/blob/main/springboot-servicio-oauth/src/main/java/com/formacionbdi/microservicios/app/oauth/SpringbootServicioOauthApplication.java>

## Gateway zull: autorización

pom.xml

<https://github.com/theinsideshine/microservicios-oauth/blob/main/springboot-servicio-zuul-server/pom.xml>